NAG Library Routine Document

F08ZTF (ZGGRQF)

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

F08ZTF (ZGGRQF) computes a generalized RQ factorization of a complex matrix pair (A, B), where A is an m by n matrix and B is a p by n matrix.

2 Specification

```
SUBROUTINE F08ZTF (M, P, N, A, LDA, TAUA, B, LDB, TAUB, WORK, LWORK, INFO)

INTEGER                 M, P, N, LDA, LDB, LWORK, INFO
COMPLEX (KIND=nag_wp)   A(LDA,*), TAUA(min(M,N)), B(LDB,*), TAUB(min(P,N)),        &
                        WORK(max(1,LWORK))
```

The routine may be called by its LAPACK name ***zggrqf***.

3 Description

F08ZTF (ZGGRQF) forms the generalized RQ factorization of an m by n matrix A and a p by n matrix B

$$A = RQ, \qquad B = ZTQ,$$

where Q is an n by n unitary matrix, Z is a p by p unitary matrix and R and T are of the form

$$R = \begin{cases} \begin{array}{c} \\ m \end{array}\!\!\begin{array}{c} \begin{array}{cc} n-m & m \end{array} \\ \begin{pmatrix} 0 & R_{12} \end{pmatrix} \end{array}; & \text{if } m \leq n, \\[2ex] \begin{array}{r} \\ m-n \\ n \end{array}\!\!\begin{array}{c} n \\ \begin{pmatrix} R_{11} \\ R_{21} \end{pmatrix} \end{array}; & \text{if } m > n, \end{cases}$$

with R_{12} or R_{21} upper triangular,

$$T = \begin{cases} \begin{array}{r} \\ n \\ p-n \end{array}\!\!\begin{array}{c} n \\ \begin{pmatrix} T_{11} \\ 0 \end{pmatrix} \end{array}; & \text{if } p \geq n, \\[2ex] \begin{array}{c} \\ p \end{array}\!\!\begin{array}{c} \begin{array}{cc} p & n-p \end{array} \\ \begin{pmatrix} T_{11} & T_{12} \end{pmatrix} \end{array}; & \text{if } p < n, \end{cases}$$

with T_{11} upper triangular.

In particular, if B is square and nonsingular, the generalized RQ factorization of A and B implicitly gives the RQ factorization of AB^{-1} as

$$AB^{-1} = \left(RT^{-1}\right)Z^{\mathrm{H}}.$$

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

Anderson E, Bai Z and Dongarra J (1992) Generalized *QR* factorization and its applications *Linear Algebra Appl. (Volume 162–164)* 243–271

Hammarling S (1987) The numerical solution of the general Gauss-Markov linear model *Mathematics in Signal Processing* (eds T S Durrani, J B Abbiss, J E Hudson, R N Madan, J G McWhirter and T A Moore) 441–456 Oxford University Press

Paige C C (1990) Some aspects of generalized QR factorizations . *In Reliable Numerical Computation* (eds M G Cox and S Hammarling) 73–91 Oxford University Press

5 Parameters

1: M – INTEGER *Input*

On entry: m, the number of rows of the matrix A.

Constraint: $\text{M} \geq 0$.

2: P – INTEGER *Input*

On entry: p, the number of rows of the matrix B.

Constraint: $\text{P} \geq 0$.

3: N – INTEGER *Input*

On entry: n, the number of columns of the matrices A and B.

Constraint: $\text{N} \geq 0$.

4: A(LDA,$*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array A must be at least $\max(1, \text{N})$.

On entry: the m by n matrix A.

On exit: if $m \leq n$, the upper triangle of the subarray $\text{A}(1:m, n-m+1:n)$ contains the m by m upper triangular matrix R_{12}.

If $m \geq n$, the elements on and above the $(m-n)$th subdiagonal contain the m by n upper trapezoidal matrix R; the remaining elements, with the array TAUA, represent the unitary matrix Q as a product of $\min(m, n)$ elementary reflectors (see Section 3.3.6 in the F08 Chapter Introduction).

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08ZTF (ZGGRQF) is called.

Constraint: $\text{LDA} \geq \max(1, \text{M})$.

6: TAUA($\min(\text{M}, \text{N})$) – COMPLEX (KIND=nag_wp) array *Output*

On exit: the scalar factors of the elementary reflectors which represent the unitary matrix Q.

7: B(LDB,$*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array B must be at least $\max(1, \text{N})$.

On entry: the p by n matrix B.

On exit: the elements on and above the diagonal of the array contain the $\min(p, n)$ by n upper trapezoidal matrix T (T is upper triangular if $p \geq n$); the elements below the diagonal, with the array TAUB, represent the unitary matrix Z as a product of elementary reflectors (see Section 3.3.6 in the F08 Chapter Introduction).

8: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08ZTF (ZGGRQF) is called.

Constraint: $\mathrm{LDB} \geq \max(1, \mathrm{P})$.

9: TAUB$(\min(\mathrm{P}, \mathrm{N}))$ – COMPLEX (KIND=nag_wp) array *Output*

On exit: the scalar factors of the elementary reflectors which represent the unitary matrix Z.

10: WORK$(\max(1, \mathrm{LWORK}))$ – COMPLEX (KIND=nag_wp) array *Workspace*

On exit: if $\mathrm{INFO} = 0$, the real part of WORK(1) contains the minimum value of LWORK required for optimal performance.

11: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08ZTF (ZGGRQF) is called.

If $\mathrm{LWORK} = -1$, a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued.

Suggested value: for optimal performance, $\mathrm{LWORK} \geq \max(\mathrm{N}, \mathrm{M}, \mathrm{P}) \times \max(nb1, nb2, nb3)$, where *nb1* is the optimal ***block size*** for the RQ factorization of an m by n matrix by F08CVF (ZGERQF), *nb2* is the optimal ***block size*** for the QR factorization of a p by n matrix by F08ASF (ZGEQRF), and *nb3* is the optimal ***block size*** for a call of F08CXF (ZUNMRQ).

Constraint: $\mathrm{LWORK} \geq \max(1, \mathrm{N}, \mathrm{M}, \mathrm{P})$ or $\mathrm{LWORK} = -1$.

12: INFO – INTEGER *Output*

On exit: $\mathrm{INFO} = 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

$\mathrm{INFO} < 0$

If $\mathrm{INFO} = -i$, argument i had an illegal value. An explanatory message is output, and execution of the program is terminated.

7 Accuracy

The computed generalized RQ factorization is the exact factorization for nearby matrices $(A + E)$ and $(B + F)$, where

$$\|E\|_2 = O\epsilon\|A\|_2 \qquad \text{and} \qquad \|F\|_2 = O\epsilon\|B\|_2,$$

and ϵ is the ***machine precision***.

8 Further Comments

The unitary matrices Q and Z may be formed explicitly by calls to F08CWF (ZUNGRQ) and F08ATF (ZUNGQR) respectively. F08CXF (ZUNMRQ) may be used to multiply Q by another matrix and F08AUF (ZUNMQR) may be used to multiply Z by another matrix.

The real analogue of this routine is F08ZFF (DGGRQF).

9 Example

This example solves the general Gauss–Markov linear model problem

$$\min_x \|y\|_2 \qquad \text{subject to} \qquad d = Ax + By$$

where

$$A = \begin{pmatrix} 0.96-0.81i & -0.03+0.96i & -0.91+2.06i & -0.05+0.41i \\ -0.98+1.98i & -1.20+0.19i & -0.66+0.42i & -0.81+0.56i \\ 0.62-0.46i & 1.01+0.02i & 0.63-0.17i & -1.11+0.60i \\ 0.37+0.38i & 0.19-0.54i & -0.98-0.36i & 0.22-0.20i \\ 0.83+0.51i & 0.20+0.01i & -0.17-0.46i & 1.47+1.59i \\ 1.08-0.28i & 0.20-0.12i & -0.07+1.23i & 0.26+0.26i \end{pmatrix},$$

$$B = \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}, \quad c = \begin{pmatrix} -2.54+0.09i \\ 1.65-2.26i \\ -2.11-3.96i \\ 1.82+3.30i \\ -6.41+3.77i \\ 2.07+0.66i \end{pmatrix} \quad \text{and} \quad d = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

The constraints $Bx = d$ correspond to $x_1 = x_3$ and $x_2 = x_4$.

The solution is obtained by first obtaining a generalized QR factorization of the matrix pair (A, B). The example illustrates the general solution process, although the above data corresponds to a simple weighted least squares problem.

Note that the block size (NB) of 64 assumed in this example is not realistic for such a small problem, but should be suitable for large problems.

9.1 Program Text

```
    Program f08ztfe

!     F08ZTF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: dznrm2, nag_wp, zgemv, zggrqf, ztrmv, ztrtrs,     &
                             zunmqr, zunmrq
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Complex (Kind=nag_wp), Parameter :: one = (1.0E0_nag_wp,0.0E0_nag_wp)
      Integer, Parameter               :: nb = 64, nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: rnorm
      Integer                          :: i, info, lda, ldb, lwork, m, n, p
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:,:), c(:), d(:),        &
                                            taua(:), taub(:), work(:), x(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: min
!     .. Executable Statements ..
      Write (nout,*) 'F08ZTF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) p, n, m
      lda = m
      ldb = p
      lwork = nb*(p+n)
      Allocate (a(lda,n),b(ldb,n),c(p),d(m),taua(min(m,n)),taub(min(p, &
        n)),work(lwork),x(n))
```

```
!     Read B, A, C and D from data file

      Read (nin,*)(b(i,1:n),i=1,p)
      Read (nin,*)(a(i,1:n),i=1,m)
      Read (nin,*) c(1:p)
      Read (nin,*) d(1:m)

!     Compute the generalized RQ factorization of (B,A) as
!     A = (0 R12)*Q,   B = Z*(T11 T12 T13)*Q, where R12, T11 and T22
!                            ( 0  T22 T23)
!     are upper triangular
!     The NAG name equivalent of zggrqf is f08ztf
      Call zggrqf(m,p,n,a,lda,taua,b,ldb,taub,work,lwork,info)

!     Compute (f1) = (Z**H)*c, storing the result in C
!             (f2)
!     The NAG name equivalent of zunmqr is f08auf
      Call zunmqr('Left','Conjugate transpose',p,1,min(p,n),b,ldb,taub,c,p, &
        work,lwork,info)

!     Putting Q*x = (y1), solve R12*w = d for w, storing result in D
!                   (w )
!     The NAG name equivalent of ztrtrs is f07tsf
      Call ztrtrs('Upper','No transpose','Non-unit',m,1,a(1,n-m+1),lda,d,m, &
        info)

      If (info>0) Then
        Write (nout,*) 'The upper triangular factor, R12, of A is singular, '
        Write (nout,*) 'the least squares solution could not be computed'
      Else

!       Form f1 - T1*w, T1 = (T12 T13), in C

!       The NAG name equivalent of zgemv is f06saf
        Call zgemv('No transpose',n-m,m,-one,b(1,n-m+1),ldb,d,1,one,c,1)

!       Solve T11*y1 = f1 - T1*w for y1, storing result in C
!       The NAG name equivalent of ztrtrs is f07tsf
        Call ztrtrs('Upper','No transpose','Non-unit',n-m,1,b,ldb,c,n-m,info)

        If (info>0) Then
          Write (nout,*) &
            'The upper triangular factor, T11, of B is singular, '
          Write (nout,*) 'the least squares solution could not be computed'
        Else

!         Copy y into X (first y1, then w)
          x(1:n-m) = c(1:n-m)
          x(n-m+1:n) = d(1:m)

!         Compute x = (Q**H)*y
!         The NAG name equivalent of zunmrq is f08cxf
          Call zunmrq('Left','Conjugate transpose',n,1,m,a,lda,taua,x,n,work, &
            lwork,info)

!         Putting w = (y2), form f2 - T22*y2 - T23*y3
!                     (y3)
!         T22*y2
!         The NAG name equivalent of ztrmv is f06sff
          Call ztrmv('Upper','No transpose','Non-unit',min(p,n)-n+m, &
            b(n-m+1,n-m+1),ldb,d,1)

!         f2 - T22*y2
          Do i = 1, min(p,n) - n + m
            c(n-m+i) = c(n-m+i) - d(i)
          End Do

          If (p<n) Then

!           f2 - T22*y2 - T23*y3
```

```
!           The NAG name equivalent of zgemv is f06saf
            Call zgemv('No transpose',p-n+m,n-p,-one,b(n-m+1,p+1),ldb, &
              d(p-n+m+1),1,one,c(n-m+1),1)

          End If

!         Compute estimate of the square root of the residual sum of
!         squares norm(r) = norm(f2 - T22*y2 - T23*y3)
!         The NAG name equivalent of dznrm2 is f06jjf
          rnorm = dznrm2(p-(n-m),c(n-m+1),1)

!         Print least squares solution x
          Write (nout,*) 'Constrained least squares solution'
          Write (nout,99999) x(1:n)

!         Print estimate of the square root of the residual sum of
!         squares
          Write (nout,*)
          Write (nout,*) 'Square root of the residual sum of squares'
          Write (nout,99998) rnorm
        End If
      End If

99999 Format (4(' (',F7.4,',',F7.4,')':))
99998 Format (1X,1P,E10.2)
    End Program f08ztfe
```

9.2 Program Data

```
F08ZTF Example Program Data

  6            4            2                        :Values of P, N and M

( 0.96,-0.81) (-0.03, 0.96) (-0.91, 2.06) (-0.05, 0.41)
(-0.98, 1.98) (-1.20, 0.19) (-0.66, 0.42) (-0.81, 0.56)
( 0.62,-0.46) ( 1.01, 0.02) ( 0.63,-0.17) (-1.11, 0.60)
( 0.37, 0.38) ( 0.19,-0.54) (-0.98,-0.36) ( 0.22,-0.20)
( 0.83, 0.51) ( 0.20, 0.01) (-0.17,-0.46) ( 1.47, 1.59)
( 1.08,-0.28) ( 0.20,-0.12) (-0.07, 1.23) ( 0.26, 0.26) :End of matrix B

( 1.00, 0.00) ( 0.00, 0.00) (-1.00, 0.00) ( 0.00, 0.00)
( 0.00, 0.00) ( 1.00, 0.00) ( 0.00, 0.00) (-1.00, 0.00) :End of matrix A

(-2.54, 0.09)
( 1.65,-2.26)
(-2.11,-3.96)
( 1.82, 3.30)
(-6.41, 3.77)
( 2.07, 0.66)                                           :End of vector C

( 0.00, 0.00)
( 0.00, 0.00)                                           :End of vector D
```

9.3 Program Results

```
 F08ZTF Example Program Results

 Constrained least squares solution
 ( 1.0874,-1.9621) (-0.7409, 3.7297) ( 1.0874,-1.9621) (-0.7409, 3.7297)

 Square root of the residual sum of squares
   1.59E-01
```